Test's questions

Question 1:

Design a circuit using PROM that has four inputs, the inputs represent a 4-bit binary number, the output of the circuit represents the BCD equivalent of the binary inputs. Tabulate the PROM truth table, then draw the PROM circuit implementation, showing the fuse mapping. **[20 Points]**.

BCD - outputs

BCD ⇒ 8-4-2-1 system

let us take the largest number = 1111 = (15)10 ⇒ (

⇒ (0001 0101)BCD

1 5

∴ the output must be 8-bits

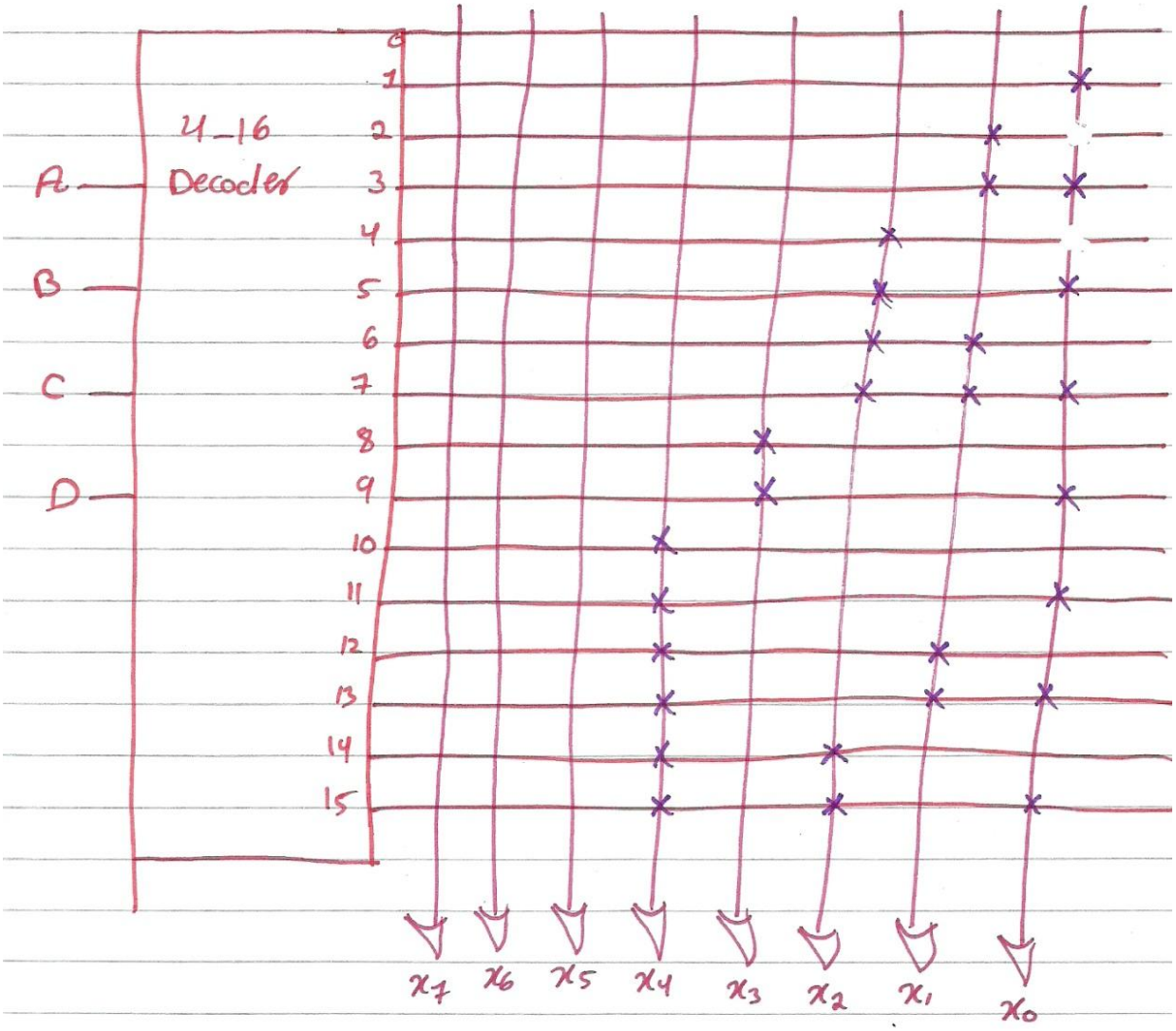
A	B	C	D	x_7	x_6	x_5	x_4	x_3	x_2	x_1	x_0	
0	0	0	0	0	0	0	0	0	0	0	0	(00)
0	0	0	1	0	0	0	0	0	0	0	1	(01)
0	0	1	0	0	0	0	0	0	0	1	0	(02)
0	0	1	1	0	0	0	0	0	0	1	1	(03)
0	1	0	0	0	0	0	0	0	1	0	0	(04)
0	1	0	1	0	0	0	0	0	1	0	1	(05)
0	1	1	0	0	0	0	0	0	1	1	0	(06)
0	1	1	1	0	0	0	0	0	1	1	1	(07)
1	0	0	0	0	0	0	0	1	0	0	0	(08)
1	0	0	1	0	0	0	0	1	0	0	1	(09)
1	0	1	0	0	0	0	1	0	0	0	0	(10)
1	0	1	1	0	0	0	1	0	0	0	1	(11)
1	1	0	0	0	0	0	1	0	0	1	0	(12)
1	1	0	1	0	0	0	1	0	0	1	1	(13)
1	1	1	0	0	0	0	1	0	1	0	0	(14)
1	1	1	1	0	0	0	1	0	1	0	1	15)

* Number of OR gates = Number of outputs = 8 O/P

* Number of AND gates = Number of inputs = 2^4 = 16 I/P

2

*Another way: Draw a block of a Decoder



Question 2 (10 points)

Design the Logic Unit shown in Figure (1) using the components of AND, OR, XOR, NOR, and MUX2*1. Write the entities of each component, and then write the complete entity and architecture pair of the Logic Unit, name any required signals.

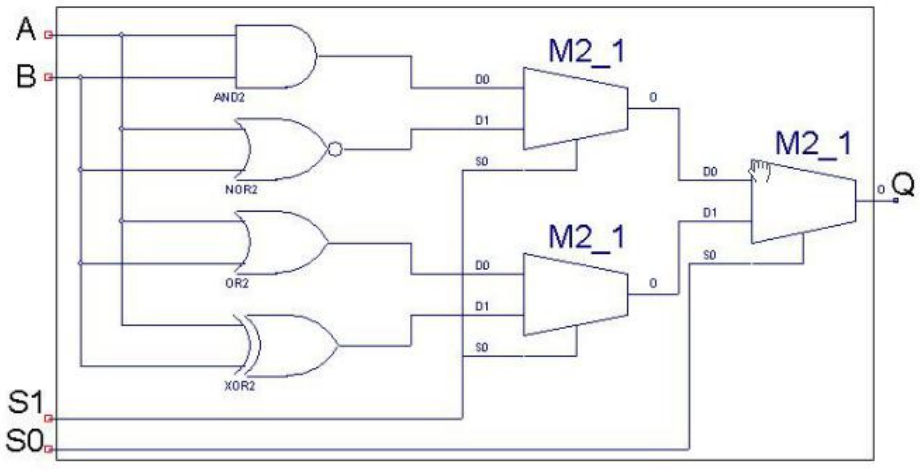


Figure (1)

```
entity my_and_gate is
   Port ( a,b : in  STD_LOGIC;
          c : out  STD_LOGIC);
end my_and_gate;

architecture Behavioral of my_and_gate is
begin
c <= a and b ;
end Behavioral;
```

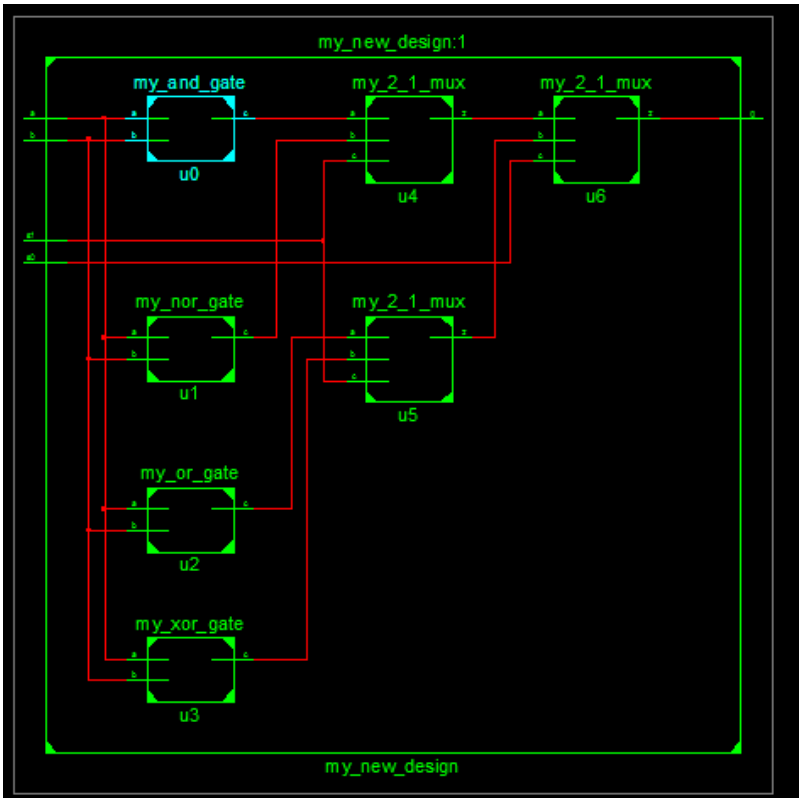
```
entity my_nor_gate is
   Port ( a,b : in  STD_LOGIC;
          c : out  STD_LOGIC);
end my_nor_gate;

architecture Behavioral of my_nor_gate is
begin
c<= not ( a or b );
end Behavioral;
```

```
entity my_or_gate is
   Port ( a,b : in  STD_LOGIC;
          c : out STD_LOGIC);
end my_or_gate;

architecture Behavioral of my_or_gate is
begin
c <= a or b ;
end Behavioral;
```

```
entity my_xor_gate is
   Port ( a,b : in  STD_LOGIC;
          c : out  STD_LOGIC);
end my_xor_gate;

architecture Behavioral of my_xor_gate is
begin
c<= ( a and not b ) or ( not a and b );
end Behavioral;
```

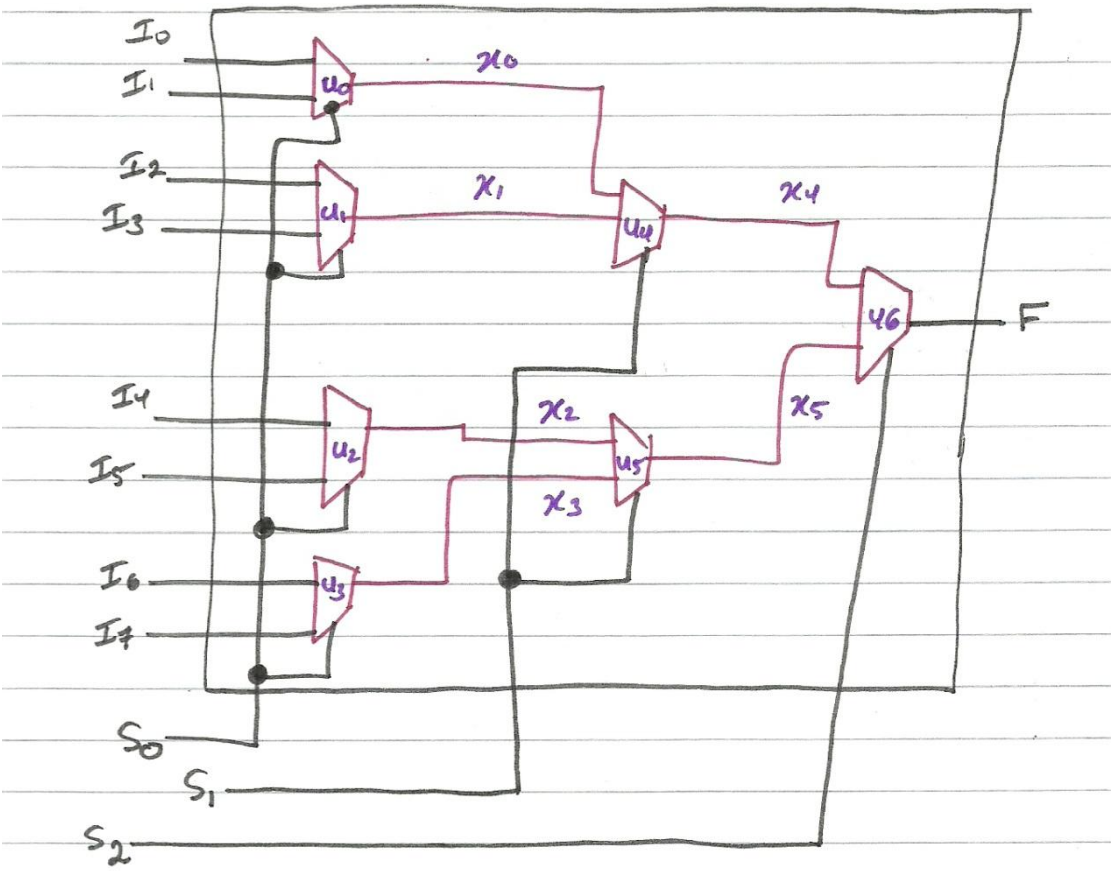
```
entity my_2_1_mux is
port ( a,b,c: in std_logic;
       z : out std_logic );
end my_2_1_mux;

architecture Behavioral of my_2_1_mux is
begin
z<= ( not c and a ) or ( c and b );
end Behavioral;
```

```
entity my_new_design is
   Port ( a : in  STD_LOGIC;
          b : in  STD_LOGIC;
          s1 : in  STD_LOGIC;
          s0 : in  STD_LOGIC;
          q : out  STD_LOGIC);
end my_new_design;

architecture Behavioral of my_new_design is

component my_and_gate is
   Port ( a : in  STD_LOGIC;
          b : in  STD_LOGIC;
          c : out  STD_LOGIC);
end component;

component  my_or_gate is
   Port ( a : in  STD_LOGIC;
          b : in  STD_LOGIC;
          c : out STD_LOGIC);
end  component ;

component my_nor_gate is
   Port ( a : in  STD_LOGIC;
          b : in  STD_LOGIC;
          c : out STD_LOGIC);
end component;

component my_xor_gate is
   Port ( a : in  STD_LOGIC;
          b : in  STD_LOGIC;
          c : out  STD_LOGIC);
end component;

component my_2_1_mux is
port ( a,b,c: in std_logic;
       z : out std_logic );
end component;


signal D0 , D1, D2,D3  , x1 , x0: std_logic;
begin
u0 : my_and_gate port map ( a, b , D0 );
 u1: my_nor_gate port map ( a, b , D1 );
 u2 : my_or_gate port map ( a, b , D2 );
 u3 : my_xor_gate port map ( a, b , D3 );
u4 : my_2_1_mux port map ( D0, D1 , s1 , x0 );
u5 : my_2_1_mux port map ( D2, D3 , s1,x1  );
u6 : my_2_1_mux port map ( x0, x1 , s0 , q);
end Behavioral;
```



Question 2

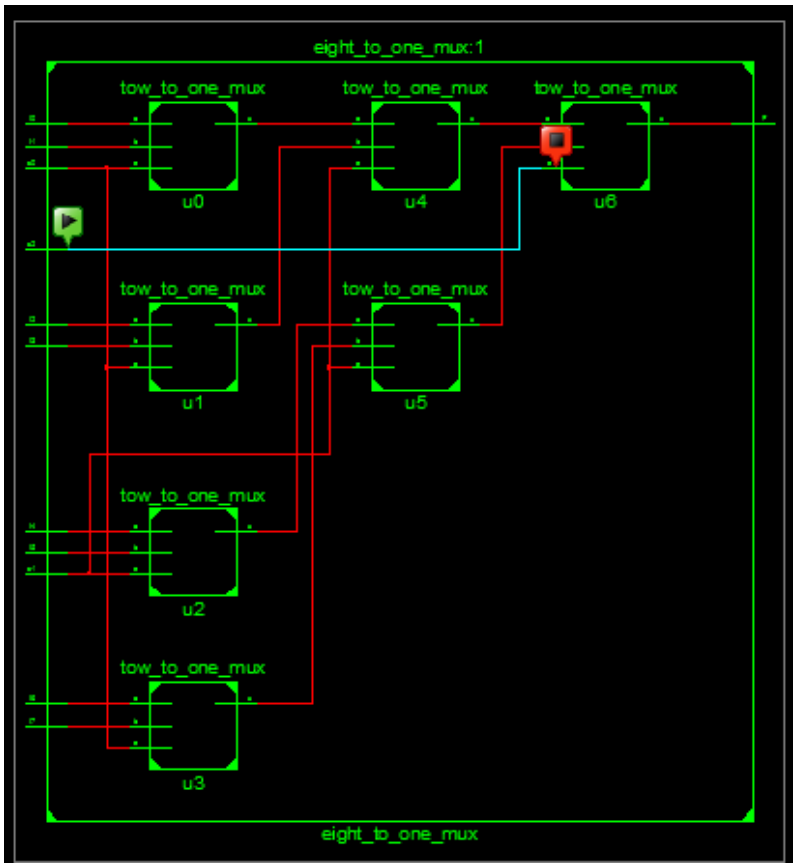
Design by using the components of 2-to-1 MUXs only, an 8-to-1 MUX. First draw a block diagram for the 8-to-1 MUX using 2-to-1 muxs, then write an entity-architecture pair for the 2-to-1 MUX, and then write an entity-architecture pair for the 8-to-1 MUX, using 2-to-1 MUXs as components, and do the port mapping for the components. **[25 Points]**




```
entity tow_to_one_mux is
   Port ( a : in  STD_LOGIC;
          b : in  STD_LOGIC;
          c : in  STD_LOGIC;
          z : out  STD_LOGIC);
end tow_to_one_mux;

architecture Behavioral of tow_to_one_mux is
begin
z<= ( not c and a ) or ( c and b );
end Behavioral;
```

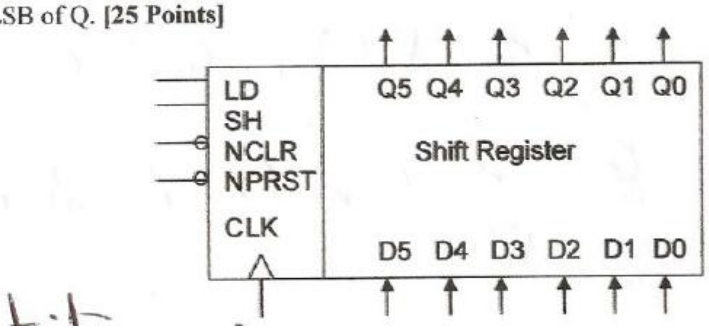
```
entity eight_to_one_mux is
   Port ( I0,I1,I2,I3,I4,I5,I6,I7,s0,s1,s2 : in  STD_LOGIC;
 F : out  STD_LOGIC);
end eight_to_one_mux;

architecture Behavioral of eight_to_one_mux is

component tow_to_one_mux is
   Port ( a,b,c : in  STD_LOGIC;
z : out  STD_LOGIC);
end component;
```

```
signal x0,x1,x2,x3,x4,x5 : std_logic;

begin
u0 : tow_to_one_mux  port map (I0, I1,s0,x0 ) ;
u1 : tow_to_one_mux  port map (I2,I3,s0,x1 ) ;
u2 : tow_to_one_mux  port map (I4,I5,s1,x2 ) ;
u3 : tow_to_one_mux  port map (I6,I7,s0,x3 ) ;
u4 : tow_to_one_mux  port map ( x0,x1,s1,x4) ;
u5 : tow_to_one_mux  port map ( x2,x3,s1,x5) ;
u6 : tow_to_one_mux  port map (x4,x5,s2,F ) ;
end Behavioral;
```



Question 3

Design by writing a VHDL entity-architecture pair a 6-bit shift register with asynchronous clear and preset, the register has the following inputs, LD, SH, NCLR, NPRST, CLK and D; the output of the register is Q. Where D and Q are 6-bits and D represents the parallel input. If NCLR=0, the output will be cleared. If NPRST=0, the output will be set to ones. When LD=1, the D input is loaded into the register, when SH=1, Q is shifted one bit to the right and D(0) is shifted to the MSB of Q, otherwise Q will be shifted one bit to the left and D(0) will be shifted to the LSB of Q. **[25 Points]**




```
entity my_shift_reg is
   Port ( ld : in  STD_LOGIC;
          sh : in  STD_LOGIC;
          Nclr : in  STD_LOGIC;
          Nprst : in  STD_LOGIC;
          clk : in  STD_LOGIC;
          D : in  STD_LOGIC_vector ( 5 downto 0 );
          Q : out  STD_LOGIC_vector ( 5 downto 0 ));
end my_shift_reg;

architecture Behavioral of my_shift_reg is
signal qint : STD_LOGIC_vector ( 5 downto 0 );
begin
process ( clk , Nclr , Nprst )
begin
if ( Nclr= '0') then
qint <= "000000";
elsif ( Nprst = '0' ) then
qint <= "111111";
elsif ( clk'event and clk='1') then
if ( ld='1') then
qint <= D;
elsif ( sh='1' ) then
qint <= D(0) & qint ( 5 downto 1 );
else
qint <=  qint ( 4 downto 0 )&D(0) ;
end if ;
end if;
end process;
Q<= qint;
end Behavioral;
```